ASSOCIATION FOR AUTOMATED REASONING
NEWSLETTER

No. 26 June 1994

From the AAR President, Larry Wos...

This issue has something for everyone.

- For the serious researcher, we offer summaries of recent workshops and a list of conferences to prepare papers for.
- For the new Ph.D., we announce the newly established Sacks prize in mathematical logic.
- For those inclined to more leisurely activities, we suggest some new books to read on a hot summer evening.
- For those "in between," we report on theorem proving reduced to a game.
- Finally, for the daring, we present a challenge with, yes, the offer of real money (though not at the lottery level).

Nominations for the Sacks Prize

The Sacks prize, for the best recent doctoral dissertation in mathematical logic, will be awarded for the first time in December of 1994. This prize was established to honor Professor Gerald Sacks for his unique contribution to mathematical logic, particularly as advisor to a large number of excellent Ph.D. students.

Eligibility: Doctorate completed between January 1, 1993, and September 30, 1994.

Nominations: Nominations are made by the thesis advisor and consist of name and birthdate of student, title and 1–2 page description of dissertation, date and location where the doctorate was awarded, and letter of recommendation from the advisor. Send nominations by September 1, 1994, to Sacks Prize Committee, Room 2-236, Department of Mathematics, M.I.T., Cambridge, MA 02139 USA; or by e-mail (encouraged) to saxprize@math.mit.edu.

For further information, contact Carla Kirmani at the above address.

Theorem Proving Reduced to a Game

S. Lecchi, F. Buffoli, and G. Degli Antoni
C. S. Dept., State Univ. of Milan (Italy)
buffoli@hermes.mc.dsi.unimi.it

As part of a dissertation, we have been developing a game (to be implemented on computer) whose rules should respect a form of logical deduction. After having studied many deduction methods, in particular those regarding proving or refuting theorems of first-order logic, we have focused on methods in which deduction is represented by graphs. In fact, graphic representation of deduction is especially well suited to form the basis of a game because one can give an immediate picture of the situation. We have chosen the simple method proposed by Peter B. Andrews in the article "Refutation by Matings," which appeared in August 1976 in *IEEE Transactions on Computers*.

The game structure is similar to the Oriental game "Taipei," in which the player has to pair different cards bearing the same picture in respect of some simple rules. In the "Mate" game, theorem clauses are represented by rows of cards of two colors: red and green. Each card represents a literal, and its color indicates whether the literal is negated or not. The player's goal is to pair cards of different colors (i.e., one literal negated with one not negated) so that every card in the game has been used in at least one pair.

Not all cards of different colors can be paired with each other, because the literals represented must be unifiable and the connection must respect the rules illustrated by Andrews in his article. Each time the player tries to pair two cards, the program checks to see whether the pair satisfies all conditions. If so, it changes the shape of the cards from squares to circles to indicate that those cards have been paired.

While the player pairs cards, the program builds the deduction graph (Andrews calls this graph "Mating") interpreting the player's actions. Each time a connection is done, the program creates the corresponding arc in the graph, and the literals involved are unified. This is done by instantiating appropriately eventual variables. If the player succeeds in pairing all cards, the final graph will respect all conditions illustrated by Andrews. This means that the graph can be regarded as a refutation of the theorem represented by the colored cards. At the end one gets a set of ground clauses (or a result in which there appears a variable whose value is not important and that can be instantiated with a random value, thereby producing a set of ground clauses) for which the same refutation holds.

For completeness, the program incorporates an automated theorem prover module that can help the player who tries to prove a theorem. Furthermore, it can prove automatically a theorem.

The strategy used to build the graph is similar to a hypothetical linear resolution applied to graph construction and is based on the procedure illustrated by Andrews. The deduction graph built by the player or the computer can be translated to resolution. This capability can be used to illustrate some aspects of automated theorem proving and can be used by a professor in order to explain basic concepts of formal deduction.

Moreover, this program allows even those who do not understand theorem proving to refute

theorems. This is because the game can be played ignoring the correspondence between cards and literals of a theorem.

References

1. Peter B. Andrews, Refutation by matings, *IEEE Transactions on Computers,* August 1976.

2. Peter B. Andrews, Theorem proving via general matings, *ACM Journal,* vol. 28-2, 1981.

3. Wolfgang Bibel, *Automated Theorem Proving,* 1987.

Challenge in Group Theory

Larry Wos
Argonne National Laboratory, Argonne, Illinois
wos@mcs.anl.gov

As so many of you know, I enjoy receiving and issuing challenges for automated reasoning programs. Here, I offer a challenge that focuses on group theory and on lattices. For this theorem, I thank Ingo Dahn, whom I met at a recent workshop held at Argonne, a workshop devoted to the laudable QED project (reported later in this newsletter). I shall state the theorem in mathematical terms, then give a first-order statement, and finally give a form used by OTTER to prove the theorem.

The shortest proof known to me was obtained by a program called Discount; it has length 33. The shortest proof I have found with OTTER has length 42. I am counting in length none of the input clauses, even if demodulated before new clauses are adjoined, nor am I counting the applications of demodulation. Therefore, my use of the word length refers to the number of paramodulation steps. (I shall return to this definition of length shortly, after I make my cash offer.)

The theorem can be attacked with a program such as OTTER, a rewrite program such as RRL, Discount, or any program of your choosing. You might prefer a Prolog-based program, one based solely on complete sets of reductions, or one employing AC-unification. The choice is yours.

The challenge is twofold. First, seek a proof, any proof, unaided by the researcher. Second, find a substantially shorter proof. For the person who finds a proof with fewer than 20 applications of paramodulation, I shall pay $2; if the proof is also elegant (to be judged by me), I shall pay $4. Only the first five successes will be eligible for a cash award; the award will only be in existence through November 26, 1994.

Finally, I must ask that any candidate for the award meets *my* definition of length. For example, if AC-unification is used, then each hidden use of associativity or of commutativity must be counted as a paramodulation step.

Here is the theorem under discussion, in three versions as promised.

Mathematical Statement

The theorem asks one to prove that, for each element x in a group, x is equal to the product of its *positive part* $pp(x)$ and its *negative part* $np(x)$, where 1 is the identity of the group, $pp(x)$ is the union of x and 1, and $np(x)$ is the intersection of x and 1.

First-Order Statement

```
-(a = *(pp(a),np(a))).
(all X (all Y (all Z (*(X,*(Y,Z)) = *(*(X,Y),Z))))).
(all X (*(1,X) = X)).
(all X (*(X,1) = X)).
(all X (*(i(X),X) = 1)).
(all X (*(X,i(X)) = 1)).
(i(1) = 1).
(all X (i(i(X)) = X)).
(all Y (all X (i(*(X,Y)) = *(i(Y),i(X))))).
(all X (n(X,X) = X)).
(all X (u(X,X) = X)).
(all Y (all X (n(X,Y) = n(Y,X)))).
(all Y (all X (u(X,Y) = u(Y,X)))).
(all X (all Y (all Z (n(X,n(Y,Z)) = n(n(X,Y),Z))))).
(all X (all Y (all Z (u(X,u(Y,Z)) = u(u(X,Y),Z))))).
(all X (all Y (u(n(X,Y),Y) = Y))).
(all X (all Y (n(u(X,Y),Y) = Y))).
(all Y (all X (all Z (*(X,u(Y,Z)) = u(*(X,Y),*(X,Z)))))).
(all Y (all X (all Z (*(X,n(Y,Z)) = n(*(X,Y),*(X,Z)))))).
(all Y (all Z (all X (*(u(Y,Z),X) = u(*(Y,X),*(Z,X)))))).
(all Y (all Z (all X (*(n(Y,Z),X) = n(*(Y,X),*(Z,X)))))).
(all X (pp(X) = u(X,1))).
(all X (np(X) = n(X,1))).
```

OTTER Statement

```
x = x.
(x*y)*z = x* (y*z).
1*x=x.
x*1=x.
i(x)*x=1.
x*i(x)=1.
i(1)=1.
i(i(x))=x.
i(x*y)=i(y)*i(x).
n(x,x)=x.
u(x,x)=x.
n(x,y)=n(y,x).
u(x,y)=u(y,x).
n(x,n(y,z))=n(n(x,y),z).
u(x,u(y,z))=u(u(x,y),z).
u(n(x,y),y)=y.
n(u(x,y),y)=y.
x*u(y,z)=u(x*y,x*z).
x*n(y,z)=n(x*y,x*z).
u(y,z)*x=u(y*x,z*x).
n(y,z)*x=n(y*x,z*x).
pp(x)=u(x,1).
np(x)=n(x,1).
pp(a)*np(a)!=a.
```

Books of Potential Interest to Our Readers

1. *Integrating Rules and Connectionism for Robust Commonsense Reasoning,* R. Sun, Wiley, 1994. Introduces a new approach to the persistent problem in AI of capturing the flexible and robust nature of commonsense reasoning.

2. *From Logic to Logic Programming,* K. Doets, MIT Press, 1994. Discusses resolution in the versions for three logical formalisms: propositional logic, first-order logic, and the so-called Horn-fragment of first-order logic.

3. *Simply Logical: Intelligent Reasoning by Example,* P. Flach, Wiley, J. C. Baltzer AG, Science Publishers, 1994. Deals with methods to implement intelligent reasoning by means of Prolog programs.

4. *Disjunctive Logic Programming*, J. Lobo, D. Loveland, and A. Rajaskar, eds., J. C. Baltzer AG, Science Publishers, 1994. Covers topics such as negation as failure, propositional semantics, and the foothold refinement.

5. *Artificial Intelligence and Mathematics III,* L. Joskowicz, F. Hoffman, and J.-L. Lassez, eds., J. C. Baltzer AG, Science Publishers, 1994. Includes propositional truth maintenance systems, preference logics, SDL–a stochastic algorithm for learning decision lists with limited complexity, and an adaptive reasoning approach towards efficient ordering of composite hypothesis.

6. *Mathematics of Modality*, R. Goldblatt, CSLI Lecture Notes No. 43, CSLI Publications; distributed by the University of Chicago Press, 1993. Collects a number of the author's papers on modal logic, beginning with his doctoral thesis about the duality between algebraic and set-theoretic models, and including two completely new articles, one on infinitary rules of inference, and the other about recent results on the relationship between modal logic and first-order logic.

7. *A Logical Approach to Discrete Math,* David Gries and Fred B. Schneider, Springer-Verlag, 1994. Aims to teach the use of logic in applications in computer science and mathematics, using an approach that begins with a solid introduction to formal proofs, and then turns to conventional discrete-maths topics (with equational logic providing the underpinnings).

8. *The Art of Prolog,* Leon Sterling and Ehud Shapiro, 2nd ed., The MIT Press, 1994. Similar to the first book but with updated discussion of recent research results, an expanded list of references, and more advanced exercises, and a heavily revised chapter on advanced programming techniques.

If you are interested in writing a book review of any of these items, please contact pieper@mcs.anl.gov.

Announcements and Calls for Papers

The 7th Banff Higher Order Workshop

This year's Banff Higher Order Workshop is based on the theme of "Logics for Concurrency." It will take place at the Banff Centre, Banff, Alberta, Canada, on August 27 - Sept. 3, 1994. The workshop is intended principally for graduate students and researchers interested in concurrency theory and/or temporal logics who wish a jump start into selected central themes. The lectures will include interaction categories, automated temporal reasoning about reactive systems, algorithms for normed processes, modal and temporal logics for processes, and automata-theoretic approach to program specification and verification. For further information, contact Graham Birtwistle, Department of Computer Science, University of Calgary, Calgary T2N 1N4, CANADA. tel: +1 (403) 220 6055. fax: +1 (403) 284 4707. net: graham@cpsc.ucalgary.ca.

International Conference on Rewriting Techniques and Applications

RTA-95 will take place on April 5–7, 1995, in Kaiserslautern, Germany. Topics include term rewriting systems, symbolic and algebraic computation, constrained rewriting and deduction, equational programming languages, string and graph rewriting, completion techniques, rewrite-based theorem proving, unification and matching algorithms, conditional and higher-order rewriting, constraint solving, architectures for rewriting, and parallel/distributed rewriting and deduction. Six copies of a full draft paper of no more than fifteen double-spaced pages are due no later than October 7, 1994; electronic submission in Postscript form is encouraged. Send submissions to the program chair, Jieh Hsiang, RTA95, Dept. of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan. Telephone: +886 2 362-2704. Fax: +886 2 362-8167. Internet: rta95@csie.ntu.edu.tw.

International Conference on Typed Lambda Calculi and Applications

TLCA will be held on April 10–12, 1995, in Edinburgh, Scotland. Topics include proof theory of type systems, logic and type systems, typed lambda calculi as models of (higher order) computation, semantics of type systems, proof verification via type systems, type systems of programming languages, and typed term rewriting systems. The deadline for submissions is September 8, 1994. Electronic submission (Postscript only) is preferred; hard copy (6 copies required) will also be accepted. Send to TLCA Secretariat, Professor M. Dezani, Universita di Torino, Dipartimento di Informatica, Corso Svizzera, 185, 10149 Torino, ITALY. Tel: 39-11-7429232. Fax: 39-11-751603. E-mail: dezani@di.unito.it. Papers should not exceed 15 standard pages and should be accompanied by a one-page abstract. Accepted papers will be published in the Springer-Verlag *Lecture Notes in Computer Science* series.

International Conference on the Mathematics of Program Construction

The International Conference on the Mathematics of Program Construction will take place on July 17–21, 1995, at Kloster Irsee, Germany. The conference emphasis is on the combination of conciseness and precision in calculational techniques for program construction. Typical areas are formal specification of sequential and concurrent programs; constructing implementations to meet specifications; program transformation; program analysis; and program verification. Submissions are due December 1, 1995. The proceedings will be published in Springer-Verlag *Lecture*

Notes in Computer Science series. For futher information, write to Professor Dr. B. Moeller (MPC'95), Institut fur Mathematik, Universitaet Augsburg, D-86135 Augsburg, Germany. E-mail: moeller@uni-augsburg.de. Fax: +49 821 598 2200.

Logic Colloquium 1995

On August 10–17, 1995, in Israel, the Logic Colloquium will take place. The conference will focus mainly on set theory, model theory, recursion theory and proof theory, and their mutual interaction and on logical aspects of computer science and linguistics. Authors are invited to submit abstracts not later than April 30, 1995. Correspondence should be addressed to logics95@cs.technion.ac.il. Logic Colloquium 95, Yvonne Sagi, Department of Computer Science, Technion–Israel Institute of Technology, 32000 Haifa, Israel.

New Series: Lecture Notes in Logic

"Logic" has come to describe the interplay between language and reference, syntax and semantics. The quest to understand and apply logic can be traced back to antiquity. Currently, we are within an explosion of new ideas and questions. Many of these are pragmatically motivated and have immediate application.

The *Lecture Notes in Logic* provide a unique forum for the discussion of topics in which logic plays a critical role. The series will draw topics from areas within which logic has a traditional place, for example within mathematics and philosophy, together with topics from within areas in which logic has an emerging importance, for example within computer science and linguistics. Further, the *Lecture Notes in Logic* will provide for the timely publication needed in these areas.

Series Editors: K. Fine, J.-Y. Girard, A. Lachlan, T. Slaman, H. Woodin.

Publisher: Springer-Verlag. Springer contact: peters@springer.de

Volume 1: J. R. Schoenfield, "Recursion Theory"; *Volume 2:* J. Ooikkonen and J. Vaeaenan, "Logic Colloquium '90"; *Volume 3:* W. Mitchell and J. Steel, "Fine Structure and Iteration."

Kluwer Joins the Network

Kluwer Academic Publishers has recently made two important additions to their services:

- The *Journal of Automated Reasoning* is now accepting LaTeX input. A special style file has been developed and is available by contacting GREGOOR.MARTENS@WKAP.nl or pieper@mcs.anl.gov.
- Information on Kluwer's artificial intelligence, mathematics, and linguistic journals (among others) is available from anonymous ftp server `ftp.std.com`, directory `Kluwer/journals`.

The QED Workshop

Gail W. Pieper
Argonne National Laboratory, Argonne, Illinois

On May 18–20, 1994, Argonne National Laboratory hosted the QED Workshop. The workshop was supported by special funding from the Office of Naval Research. Approximately 30 researchers from around the world assembled to consider whether it is desirable and feasible to build a proof-checked encyclopedia of mathematics, with an associated facility for theorem proving and proof checking. Among the projects represented were the Argonne/OTTER group, Mizar, Nqthm, HOL, Eves, MathPert, ILF, NuPrl, Coq, RRL, and Imps.

The structure of the workshop was intentionally kept informal; no formal presentations were given. Moreover, the discussions themselves were focused on nontechnical issues—potential customers, philosophy, linkages with other symbolic and numerical systems. The result was lively discussion, often sharp disagreement, about a variety of political, sociological, and aesthetic questions involved in organizing such a major undertaking as the QED project.

Among the topics discussed were the objectives of QED, QED components, interfaces, object language and basic theory, meta-theory, and libraries and tools.

The most important conclusion drawn at the QED Workshop was that *QED is an idea worthy pursuing*, a statement with which virtually all the participants agreed. To achieve this objective will require several important changes.

- Common terminology is needed. The numerous projects represented at the workshop have been developed in relative isolation so far. In these isolated projects, people tended to name same things differently and to give the same name to different functions.
- There is also a need to agree on the name for the area into which QED-like activities fall. This will be the starting point of a QED scientific community.
- The experience collected by separate teams, no matter how impressive, is still too small to extrapolate into the fully fledged QED. More of such small-scale experience is required.

The workshop participants also drew several conclusions about future research directions, including the development of specific exercises (e.g., implementing inductive definitions, as in Coq, in the framework of set theory, as in Mizar) and the start of bilateral projects (e.g., a collaboration between HOL and Mizar).

Additional information on the workshop discussions and on suggestions for the start of a QED scientific community are given in a report on the QED Workshop, available from pieper@mcs.anl.gov.

Computational Logic 1994 Research Review

William D. Young
Computational Logic, Inc.
1717 West 6th Street, Suite 290
Austin, Texas 78703

Computational Logic, Inc. (CLI), is a small employee-owned company founded in 1983 by computer scientists associated with the Institute for Computer Science and Computer Applications at the University of Texas at Austin. The company currently employees 17 researchers and 6 members of the support staff. CLI specializes in advanced research and development in mathematical modeling of digital hardware and software systems. ("Mathematical modeling" in CLI parlance encompasses what is commonly referred to by the less precise term "formal methods." CLI regards "formal methods" as ambiguous and not adequately suggestive of its principal research activity, the application of rigorous mathematical reasoning to achieve reliable digital systems.) Notable examples of CLI's research include the CLI "stack"—a hierarchically-integrated collection of verified components including a compiler for a simple high-level language (Micro-Gypsy), assembler and linker for the Piton assembly-level language, and the FM9001 microprocessor—the verified Kit kernel, and many others. The company distributes and supports the Boyer-Moore Nqthm theorem prover and its Kaufmann interactive enhancement and, until recently, maintained the Gypsy Verification Environment.

CLI bi-annually hosts a public review to showcase its research and to obtain feedback from professional colleagues and sponsors. The 1994 review was held April 20–22 in Austin. Approximately 60 representatives of academia, industry, and government attended the review.

CLI presented research on a variety of topics in automated reasoning and software and hardware verification. Although presentations were diverse, one consistent theme was evident. Earlier CLI tools and techniques for modeling and analyzing digital systems tended to be applied to small, special-purpose systems designed with formal analysis in mind and often ignoring many hard "real-world" issues considered difficult from a formal modeling perspective. Current CLI research is aimed at demonstrating the feasibility of applying these tools and techniques to more realistic applications, including languages, hardware modules, and system software of genuine commercial interest (Ada, C, VHDL, Mach, DSPs).

In the remainder of this note we briefly summarize the content of the research review. Additional information about any of the topics presented or a CLI publication list can be obtained from CLI (contact elster@cli.com).

Acl2: Acl2 is the proposed successor to the Boyer-Moore Nqthm theorem prover. It is under development by Bob Boyer, Matt Kaufmann, and J Moore at CLI and is expected to provide the foundation for future CLI modeling and analysis research. The logic of Acl2 is an applicative subset of Common Lisp consistent with commercial implementations. The logic is supported by a powerful proof engine.

- J Moore and Matt Kaufmann summarized the key features of the language and of the proof system.

- John Cowles (a CLI consultant) from the University of Wyoming described the development of libraries of definitions and lemmas to support arithmetic reasoning with Acl2.
- Larry Akers described his recently completed dissertation research that involves a type information reasoning system for Acl2.

A related presentation was made by Gerard Huet of INRIA (France) on the Coq theorem-proving system.

Languages: CLI researchers presented several projects related to defining the semantics of programming languages.

- Art Flatau described his recently completed dissertation research on a formally verified compiler from the Boyer-Moore Nqthm logic to the Piton assembly level language. This project included the specification of dynamic storage allocation and garbage collection.
- Mike Smith described an ongoing project to define a semantics for a subset of Ada suitable for analysis.
- Larry Akers outlined work aimed at defining a mathematical semantics for a subset of C.
- Matt Kaufmann described a technique for extracting verification conditions automatically from annotated programs.

Controllers: One of the developing application areas of CLI's mathematical modeling technology is control and associated real-time issues. Several presentations were related to this topic.

- Matt Wilding gave back-to-back presentations on: mechanically supported proof of a classic scheduling algorithms, and the modeling and verification of real-time applications written for a version of the CLI stack.
- Miren Carranza described the modeling and verification of a fuzzy controller for a simple water tank system.
- Bill Young described the modeling of a railroad gate controller using Nqthm.

System Software: A strong research thread within CLI has been verification of system software, such as the CLI stack and Bill Bevier's verification of an operating system kernel (Kit). Current research that builds on that foundation was presented.

- Bill Bevier described a temporal logic-based model of the Mach operating system kernel.
- Larry Smith outlined a plan to use the Mach model as a vehicle for testing particular Mach implementations.
- Rich Cohen described the ongoing modeling and proof of a simple separation kernel that provides separated virtual task address spaces on a uniprocessor.

- Bill Bevier described the development of a theory of noninterference-style security and the use of this theory in analyzing several simple trusted computing bases.

Hardware: CLI has a very active research program in hardware verification. Hardware related presentations included the following.

- Warren Hunt described an approach to extending CLI's research into modeling and verifying hardware to encompass analysis of digital signal processors (DSPs). Hunt and Ed Greenwood of Motorola outlined an ongoing project to model a Motorola DSP.
- Yuan Yu, formerly a student of Bob Boyer at UT and currently at DEC, described his surprisingly painless adaptation of his dissertation work modeling the Motorola 68020 user instruction set to handle the DEC Alpha processor.
- Ken Albin described validation of Yu's 68020 model by executing the specification and comparing the computed results against an actual 68020 chip.
- Larry Smith described a similar validation of CLI's formally verified FM9001 chip description against the fabricated FM9001 chip.
- Warren Hunt outlined a joint project with Bob Boyer to code a simulator for the full VHDL hardware description language in Acl2.
- David Russinoff described the definition of a semantics for a subset of VHDL intended to support mechanical proof.

DSP Design Session: The third day of the research seminar was a proprietary session devoted to the CLI/Motorola collaboration aimed at modeling and verifying portions of a Motorola commercial DSP. In addition to Warren Hunt's introduction to the project, there were presentations on the following topics.

- Fay Saydjari of the U.S. Government presented a tool suitable for developing custom netlists for hardware such as DSPs.
- Ed Greenwood of Motorola described the design of the DSP under analysis.
- Bishop Brock described his efforts in modeling the DSP within Acl2.
- Ken Albin and Warren Hunt described the modeling and formal verification of the implementation of two algorithms on the DSP: a reciprocal algorithm and a fast fourier transform.